Dynamic Scheduling System for Web Monitoring / Crawling System

By

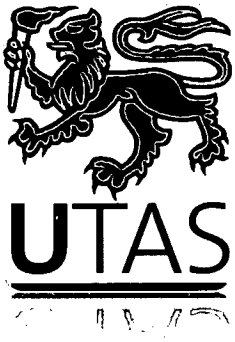
Nawaf Saleh AlDuhaim

A dissertation submitted to the School of Computing in partial fulfilment of the requirements for the degree of

Master of Computing

Supervisor

Dr. Byeong-Ho Kang



University of Tasmania

June, 2010

Dynamic Scheduling System for Web Monitoring / Crawling System

Keywords: outlier, knowledge base system, dynamic scheduling, web monitoring.

Abstract

This research studies the updating speed of web pages to provide a dynamic scheduling system for web monitoring. It combines two approaches; Multiple Classification Ripple-Down Rules (MCRDR) and Detector Constructor (DC-1). MCRDR is used to retrieve articles from web pages and classify them into folders. The DC-1 then checks if there is any unusual posting activity in these folders to inform the MCRDR to schedule a new revisiting time sooner than the originally scheduled time. This system aims to keep the user updated with fewer visiting times and less delay time between publishing (a change in web page) and collecting times (revisiting time).

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible. I would like to express my sincere gratitude to my supervisor, Associate Professor Byeong Kang. His wide knowledge and his logical way of thinking have been of great value to me. His understanding, encouraging and personal guidance have provided a good basis for the present thesis.

Besides my advisor, I would like to thank the rest of my thesis committee: Yang Sok Kim, Dr. Jacqui Hartnett and Dr. Kristy de Salas for their encouragement and insightful comments.

I am also grateful to Morag Porteous and Louise Oxley for encouraging the use of correct grammar and consistent notation in my writing.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the research.

Table of Contents

1 Introduction

Currently, the World Wide Web has a huge impact on delivering specific information in response to the user requests. It may become the most valuable resource for providing quick and relevant information about life events and education. Thus, a great range of websites have to update their web pages daily to keep track with new events or developments in any field.

With this speed of updating the pages, people may find it very difficult to cope manually when they try to visit page by page and recognize what is new since the last visit. This project aims to provide a dynamic scheduling mechanism in order to automate the revisiting time to these web pages with respect to unexpected or unusual events (such as the Olympic Games, economic crisis ...etc). Furthermore, dealing with these events may require shorter revisiting time in order to keep up with the faster updating speed of a web page.

2 Literature Review

2.1 Web Monitoring Systems (WMS)

A wide variety of web monitoring systems were created to keep up with the updating speed of a web page. Kang (2009) states that these web monitoring systems have been developed to achieve two main objectives. The first objective is detecting changes in the targeted web page without missing any information. Finding the least delay between publishing (a change in web page) and collecting time (revisiting time) is the second objective.

CONQUER, Niagra, openCQ and WebCQ are web monitoring systems that are proposed by many researchers, and sometimes called Continuous Query systems (CQ). These methods may be called the first generation of WMS that opened the gate for evaluation of the WMS. Although they are in different contexts, they are used to compare changes in objects on the web page when they do the revisiting, ignoring some of the WMS concerns such as hyperlinks, images and text (Kang et al. 2009).

Many WMS were developed in order to overcome the previous systems problems. One of these systems is Continuous Adaptive Monitoring (CAM), which is developed by (Pandey, Sandeep, Ramamritham & Chakrabarti 2003). CAM is a statistical approach that minimizes information loss by allocating limited monitoring resources across pages. CAM is more efficient and optimal than the previous methods due to the fact that it "keeps responses to continuous queries current by focusing on the problem of dynamically monitoring the sources of information relevant to the queries"(Pandey, Sandeep, Ramamritham & Chakrabarti 2003).

CAM is composed of four phases. The first phase is identifying pages relevant to a set of queries. In this phase, each returned query from a page comes with pages relevant to this page. CAM will then track and characterize relevant page changes. After that, comes the resource allocation phase, where CAM needs to minimize the weighted importance of changes that are not reported to users. The final phase is scheduling the monitoring tasks, where CAM produces a near-optimal schedule for monitoring (Pandey, S, Dhamdhere & Olston 2004).

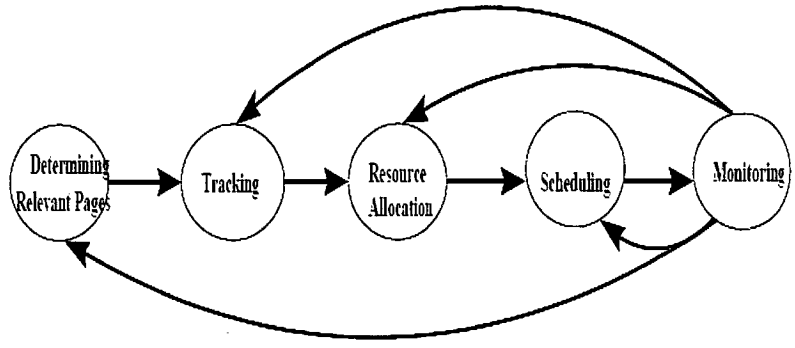


Figure 2-1: CAM Phases (Pandey, Sandeep, Ramamritham & Chakrabarti 2003)

This system may have a major impact on WMS, but it has some significant issues. Kang et al. (2009) reported that CAM failure occurs due to the fact that it cannot easily cope with bursts and does not clearly and directly model time-varying update frequencies to sources. Pandey et al. (2004) also declared that CAM is only appropriate for a narrow range of applications in which timeliness of information captured is of highest importance.

Pandey et al. (2004) present a new general-purpose Web monitoring algorithm called the Web Information Collector (WIC) that tries to overcome CAM limitations. WIC deals with a wide range of application scenarios and it performs within a factor of two of the optimal offline Web monitoring algorithm in all cases. Furthermore, it is a practical system for real-world use due to the fact that it is highly efficient and executes in an online fashion.

In contrast to CAM, which is a pull-based data source, WIC is a push-based stream that checks sources for updates at regular intervals. This system is designed to allow users to control the trade-off between timeliness (an update

notification) and completeness (a notification of all changes to an object) when bandwidth is limited. Both user preferences and the probability of updates to an object are the main basis of the WIC choice to refresh the objects (Bright, Gal & Raschid 2006; Kang et al. 2009; Pandey, S, Dhamdhere & Olston 2004). Although WIC is useful for many wide area applications such as online auctions or archiving web sources, it does not consider an important aspect of any pull-based policy, which is how to determine the probability of an update to an object (Bright, Gal & Raschid 2006; Kang et al. 2009).

There are several limitations in the previous systems that make them undesirable for this research. The first issue is these approaches do not give credit to how the user uses the monitored information: the user may not be interested in most of it. Also, the user needs to do further process such as filtering to avoid overloading. Furthermore, these systems ignore how different users value monitored information (Kang et al. 2009).

Allied to these issues, the previous systems pay no attention to an event that increases the value of posted articles on the web such as Olympic Games and economic crises.

2.2 Knowledge based systems

Knowledge-based systems (KBS) present a technique that is used to formalize and automate knowledge (Hendriks & Vriens 1999). A number of knowledge-based systems keep a record of the rules that have been applied and play back the relevant portion of that record to carry out their results. "The system examines the knowledge base to find all the rules that would have led to a conclusion". Furthermore, these systems acquire more knowledge or rules during the processing time (Davis 1986).

Kang, Compton and Preston (1995) explain many systematic knowledge based systems such as KADS, Chandrasekaran and Hayes-Roth. Regardless of how these systems are implemented, all problems are solved by given certain inputs to the systems. They will then perform closely matched outputs without any constraints on the links and parts of links between input and output that are provided by the expert.

Ripple Down Rules (RDR) is designed to overcome the above issue, due to the fact that it provides a very simple system for linking inputs and outputs to highly constrain the expert. These new links are added and corrections made in a way that does not have any negative impacts on the performance of the knowledge base (Kang, Compton & Preston 1995).

Kang (1995) defined the context in RDR as the sequence of rules that lead to a wrong conclusion (or no conclusion) when they are evaluated. The rule is added when it generates a new conclusion. This rule is evaluated only after the same sequence of rules is evaluated. The structure of the RDR's knowledge based will be a sequence of ordered rules (if ... else-if rules), with exceptions. Therefore, if the data is compatible with a certain rule, the data will take the rule conclusion unless there is exception in the rule that prevents processing this data.

Another characteristic of RDR that it allows the user to add only a valid rule; the added ruled is joined to a specific case which is called a cornerstone case. When a new rule is added and does not match with the saved cornerstone cases, a new cornerstone case will be added to join the new rule (Kang, Compton & Preston 1995).

Although RDR has succeeded in many fields such as PIERS, which is an expert system used to add clinical interpretations to chemical pathology laboratory reports, RDR has many drawbacks. One of these drawbacks is that RDR handles a single output for a given input, which clearly does not enable multiple conclusions for a given input. Another issue is that a considerable repetition of knowledge may result. This occurs when users tend to produce very general rules (Kang, Compton & Preston 1995).

2.3 Scheduling Strategies

Scheduling strategies aim to get the maximum documentation coverage with a minimum delay. With these scheduling features, the web monitoring systems would provide faster delivery, and save network and processing time for the user (Kang et al. 2009).

Most web monitoring systems that are mentioned above use static scheduling for revisiting delay. For example, CAM does some calculation on the pages time to set a feasible and optimal revisiting time (Pandey, Sandeep, Ramamritham & Chakrabarti 2003). However, CAM ignores that some web pages, such as news web pages, do not have a certain time for updating their pages if an event is happening. Therefore, when it is updating the revisiting time, it might have to do the calculations and the evaluation in each visit. As a result, this scheduling strategy would not be feasible because it will increase the processing time and it may not get the desired quality of the fast delivery.

2.3.1 Outlier Detection

Outlier detection is needed to design an appropriate dynamic scheduling strategy. (Hodge & Austin 2004) defined an "outlier as one that appears to deviate markedly from other members of the sample in which it occurs". In

other words, it is an unusual occurrence that happens during the running process. This unusual occurrence is represented as an event that would change the updating time or the revisiting time for web pages.

Currently, there are a great deal of outlier detection methodologies, and they can be categorized into four fundamental models which are statistical models, neural networks, hybrid systems and machine learning. There are no advantages of one model over another and it only depends on how an algorithm is suitable for the work (Hodge & Austin 2004) .

One of the statistical models is Grubbs' method (extreme studentized deviate). This method calculates the median as the regular situation and surrounds it with standard deviation. If the current situation exceeds the standard deviation, the method will raise the alarm, otherwise it is in the normal stage (Hodge & Austin 2004) . Grubbs' method does not require any interference from the user and it is used for real-valued data sets.

Another model of detection of outliers is the neural network. In general, neural network methodologies are non-parametric and model-based. They perform well with unseen patterns and are able to learn complex class boundaries. After training, the neural network forms a classifier. However, the entire data set has to be traversed various times to allow the network to resolve and model the data correctly. Before they are ready for the classification of new data, they require both training and testing to fine tune the network and determine threshold settings (Hodge & Austin 2004) . Nairac et al. (1999), Bishop (1994) and Japkowicz et al. (1995) are examples of this model.

The majority statistical and neural methodologies require fundamental or at the least ordinal data to allow vector distances to be determined and have no mechanism for processing categorical data with no implicit ordering. In contrast, the machine learning model does not require any prior knowledge of the data. It mostly uses decision trees which are robust, they concentrate on the salient attributes, and work well on noisy data (Hodge & Austin 2004) . There are several examples of machine learning such as John (1995), Skalak and Rissland (1990) and Arning et al. (1996).

Hybrid systems models are considered as the most recent development of outlier detection. It joins at least two of the previous models' approaches in one method. Hybridisation is used variously to overcome insufficiencies with one particular classification algorithm (Hodge & Austin 2004) . JAM system (Java agents for meta-learning) is an example when it joins five machine learning methods in one method (Stolfo et al. 1997).

3 Methodology

This research makes use of the multiple classification ripple-down rules as the knowledge based system that retrieves articles form web sites and classifies them. Classification will separate the desired article from the others. To detect events, the system employs the Detector Constructor as outlier detection to achieve the dynamic scheduling approach goals.

The Detector Constructor will look at the classified folders which contain the classified article. It will then process these folders according to whether or not an outlier is present. If one or more folders have an unusual event, the DC-1 will notify the MCRDR classifier to reschedule the visiting time for those

folders. In addition, if there is an unusual posting in one article or a few articles, the DC-1 will recommend setting the revisiting time for these articles.

3.1 The System in Theory

3.1.1 Multiple Classification Ripple-Down Rules

Multiple Classification Ripple-Down Rules is a document classification system that is used to detect unexpected events by comparing the similarity of web pages. This similarity is determined by the number of articles stored in the same category (Kang et al. 2009).

Multiple Classification Ripple-Down Rules was developed to overcome the RDR limitation. MCRDR extends RDR as it allows multiple independent classifications and it may present a basis for building a general problem solver for a range of problems other than classification. So, MCRDR has multiple cornerstone cases for a rule, compared to RDR where there is one cornerstone per rule (Kang, Compton & Preston 1995).

The MCRDR system uses an incremental learning process to build a classification knowledge base over time. This incremental method is used to cope with the continuous posting of documents and adapt to changes in the classified knowledge over time (Everts, Park & Kang 2006 ; Kang et al. 2009).

MCRDR Document Classification System

Kang et al. (2009) demonstrate that the multiple classification ripple-down rules system consists of two trees. Both of them are initially identified by the

user. The first tree is called a category tree, which used for managing the user's domain knowledge. The category tree acts in a similar way to a common folder structure and represents hierarchical relationships among categories. It is easy to maintain the category tree by using domain experts to manage a conceptual domain model through simple folder manipulation.

The other tree is called a rule tree, which is an n-array tree that is used to save the user's rule classification knowledge. In other words, the rule tree keeps a record of all rules that a user would implement to classify contents. Further, exceptions can be added to a rule, and this rule becomes a parent of these exceptions rules. As an example of these exceptions, the figure below shows that Rule 2 is an exception to Rule 1. Every rule should indicate a category in the category tree, and a rule with null indicating value will be re-marked as a stopping rule. Through the classification process, the MCRDR classifier evaluates each rule node of the knowledge base (KB).

As an example of the MCRDR classification scenario, assume that there are two documents that need to be classified with MCRDR, and each document has a set of keywords. The first document is T, which has a set of keywords {a, b, d, k}, and the other document is B, which has a set of keywords {f, s, q, r}. In advance, the MCRDR user creates the structure of the category tree and adds the rules as shown in Figure 3-1. Rule 1 has two child rules (Rule 2, Rule 3), and Rule 4 has Rule 5 as a child rule or exception.

Each rule indicates a category to classify the document. Rule 1 classifies any document that stops under it into category 1, for instance. Rules 2, 3, 4 and 5 indicate categories 2, null, 2 and 5 respectively. As the first step in the process, MCRDR maintains the two documents and checks the set keywords of each document with the parent rules, or the first level rules in the rule tree

(which are Rule 1 and Rule 4 in this example). Then, valid first level rules will open the gate to check the document with its child rules, or next level rules.

After that, MCRDR will check with the next level. If there is a satisfied rule the document will continue passing through the levels until there are no more child rules, or stay with a valid parent rule if its child rules are not satisfied. In the example, T and B document will stop at Rule 3 and Rule 5. Since Rule 3 refers to null, the documents will be placed in category 5, which satisfies Rule 5.

When the user creates the trees, he/she constructs a knowledge base, called the knowledge acquisition (KA), which is bonded with the classifying processes in a MCRDR classifier.

At the start of the system, a knowledge base has no rules. Also, there is no classification category. In this case, MCRDR classifies the document to Rule 0, which is the root that accepts any document. For example, if Figure 3-1 is assumed not to have any rules, and case 1 is obtained by the system, it will do no action or recommendation regarding the case because the system does not have any information that would lead it to take any decision. If the user adds as the next step Rule 1 and Rule 4, the case will be successfully classified to two categories, one and two, because case 1 satisfies both rules.

Case 3 is an instance of a stopping rule and the MCRDR classifier will send it to category 1. This happens because Rule 3 is classified to null and the system will recommend the parent rule category for classification. This situation occurs when the user creates the rule and forgets to specify the destination

folder from the category tree. Kang et al. (2009) show the MCRDR rule creation process as follows:

1. The user creates folders or categories in the category tree for classification.
2. The user chooses a document that he/she is interested in.
3. From the category tree, the user selects the destination folder and selects one or more keywords from the document to make a rule.
4. Depending on the keywords, the system produces document lists satisfying rules in this new rule path (Rule0 – Rule 1 – Rule 3 (new rule));
5. From the document lists, the user can discard one or more irrelevant documents. Then, the MCRDR classifier presents the difference lists instead of the case attributes.
6. The user needs to repeat steps 2-5 to list all desired rules.

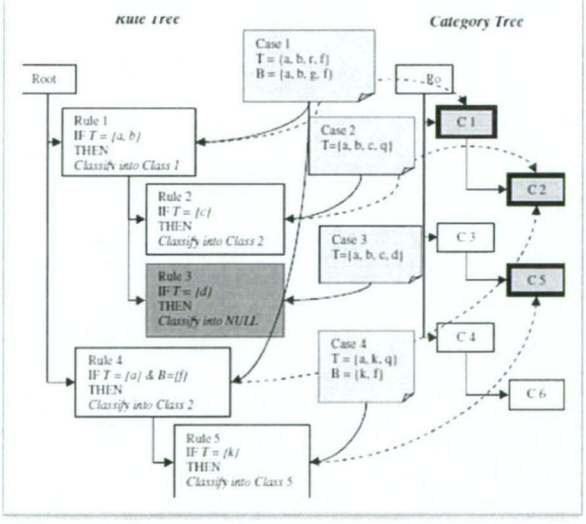


Figure 3-1: Knowledge Base of MCRDR Classifier (Kang et al.2009)

This research applies MCRDR as a knowledge based system because the MCRDR classifier enables low cost knowledge maintenance (Kang, Compton & Preston 1995). Also, it has been successfully used in a variety of situations. Another advantage is that MCRDR is easy to use and effective. Furthermore, MCRDR classifier performance has been positively evaluated by many researchers (Kang et al. 2009).

3.1.2 The Detector Constructor

The detector constructor (DC-1) is an activity monitoring system used for detecting unusual activity or news story monitoring. Its ruling based technique has many applications. For example, it can be a classifier, learning classification rules from both regular and irregular training data. Also, the ruling system has the ability to be used as a recognizer trained on regular data only, or as a rules learner to recognize changes which detect outlier activity (Fawcett & Provost 1997, 1999; Hodge & Austin 2004).

DC-1 is a rule-based system which is similar to MCRDR. The system is classified as machine learning outlier detection. This approach has been chosen due to the need for dynamic scheduling in web monitoring. Dynamic web pages keep changing and the amount of change is not fixed. For example, a news web page posts articles depending on current events. If there is an unusual event, the number of posted articles will be more than regular days. In this research, DC-1 is used to deal with these situations because it is an adaptive system.

DC-1 was chosen because it is a flexible and incremental rule-based system. Comparing to machine learning approaches such as decision trees, adding new rules may be added easily with DC-1 or rules altered without disturbing the existing rules. Unlike the decision tree, DC-1 does not require generating a complete new tree. Further, DC-1 has an efficient mechanism for processing categorical data with implicit ordering compared to most statistical and neural approaches. In addition, those approaches require basic or organized data to allow vector distances to be estimated. In contrast, DC-1 does not require any data in advance (Hodge & Austin 2004).

The Detector Constructor System

The DC-1 system consists of three stages which are: learning and selecting rules, profiling monitors, and value normalization and weighting. In the first stage, which is creating and selecting rules, the system will create rules automatically due to the fact that the DC-1 has a rule generator that will generate a great deal of rules. DC-1 has a selection algorithm that identifies a small set of general rules and these selected rules are used to construct profiling monitors (Fawcett & Provost 1997).

The second stage is profiling monitors, consisting of two steps: the profiling monitoring step and the use monitoring step. In the profiling monitoring step, the monitor measures the normal activity of a single entity and the resulting statistics are saved with the entity. In the use monitoring step, the monitor measures the daily activity for the entity and saves the resulting statistics with the entity (Fawcett & Provost 1997).

To produce these statistic outputs, there are two common monitor templates that are applied by the system. These monitors are threshold and standard deviation monitors. Threshold monitors use rule conditions on the daily basic record and count the number of conditions that satisfy the given rules and keep track of the maximum as a daily threshold for the profiling monitor. In the use monitor, the threshold monitor applies the rule conditions on a day-record, and estimates the number of conditions that satisfy the rules. Then, it compares the result with the profiling monitor to produce outputs (Fawcett & Provost 1997).

$$\text{Output} = \begin{cases} 1 \; if \; |R| > daily\ threshold \\ 0 \; otherwise \end{cases}$$

- R is the estimated number of conditions that satisfy the given rules in the day-record.

The DC-1 System also has standard deviation monitors which are sensitive monitors. They estimate the average amount of activity on an entity and the expected daily variation of that activity. They are sensitive because entities with the same average produce different values from the monitor in the profiling step if their standard deviations are different (Fawcett & Provost 1997).

In profiling monitoring, the standard deviation monitor applies the given rules to a range of basic daily records. Then, the system estimates the mean and standard deviation from the number of conditions that satisfy the given rules in these records. In use monitoring, DC-1 evaluates a day-record and gives outputs as follows:

$$\text{Output} = \begin{cases} R & if\ \sigma = 0 \\ \frac{R-\mu}{\sigma} & if\ R > \mu \\ 0 & otherwise \end{cases}$$

- R is the estimated number of conditions that satisfy the given rules in the day-record.
- μ and σ are the mean and the standard deviation respectively which are calculated in profiling monitoring (Fawcett & Provost 1997).

In web monitoring, as an example of Standard deviation monitor, assume that a rule is (Time = 10:00:00) AND (Site Name = BBC). The profiling monitor calculates the average and standard deviation of BBC articles that are posted with this date, and these articles are assigned to a folder. For instance, BBC posts 10 articles as an average and 3 as standard deviation. For this rule, these values (10, 3) will be attached to the folder. In use monitoring, if the monitor processed a time visit containing 7 BBC articles at 10, the monitor would send a zero; if the monitor detected 22 articles, it would send (22 - 10)/3 = 4. This value denotes that the account is four standard deviations above its average (profiled) usage level.

In the third stage, DC-1 combines the statistical result for each entity and compares them. The DC-1 system uses the results to identify if there is an unusual situation or not. In other words, it weights the profiling monitors outputs and combines the threshold to the outputs so that alarms may be issued with high confidence (Fawcett & Provost 1997).

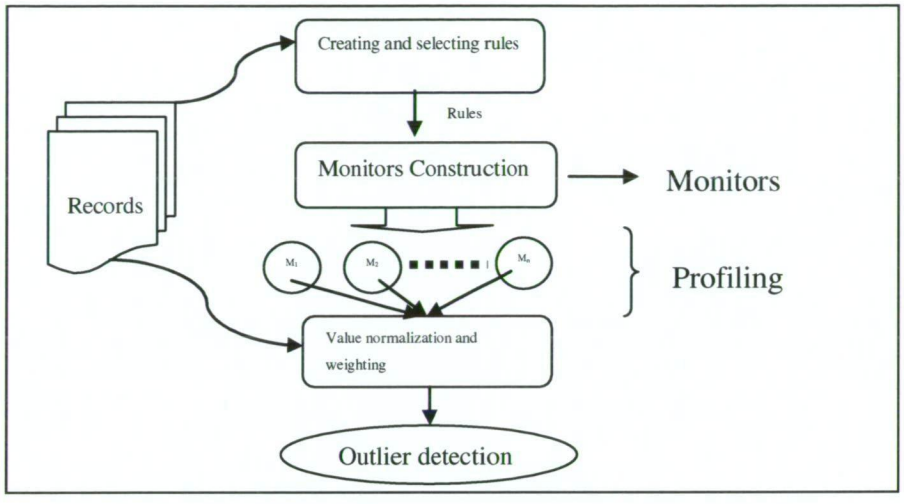
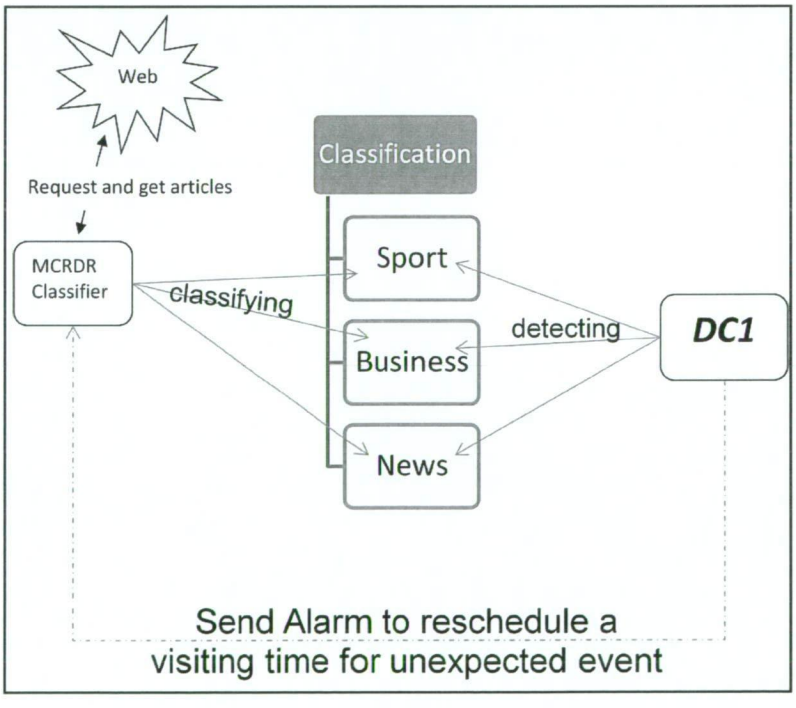


Figure 3.1.2 The Detector Constructor System (Fawcett & Provost 1997)

3.2 The System in Practice



3-2 System Structure

3.2.1 Background Information

This research extends Kang et al.' (2009) approach which uses MCRDR to retrieve articles from the World Wide Web and classify them into folders. Then, three event-driven scheduling approaches were applied to the classified folders to set the revisiting time if there an unusual or unexpected event: Top-down, Bottom-down and Random scheduling approach.

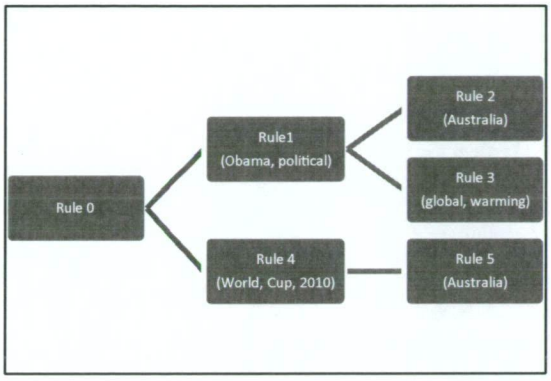
These scheduling approaches were compared with a static approach. The results showed that with small monitoring intervals there was no significant difference between a static scheduling approach and the three dynamic scheduling approaches. With large monitoring intervals however, the dynamic scheduling strategies give greater monitoring performance compared to the static approach.

These results were a simulation of the data retrieved during the Olympic event and still not tested in a real-system. This research aims to give an approach that can be used as a dynamic scheduling system which is the Detector Constructor.

Although the system is considered as an old system since it was implemented in 1997, Hodge (2004) believes that in the outlier detection field there is no advantage of one approach over another. An approach can be selected on the basis of its compatibility with the system.

3.2.2 MCRDR Classifier

The multiple classification ripple-down rules system is used as knowledge based system that retrieves data or articles from specified web pages that the user of the system is interested in. The user should insert rules that enable the



3-3 : Moving from general rules to more specific

MCRDR system to use these rules as conditions. Each rule is documented with a classification folder. In other words, the MCRDR system will classify articles in virtual folders (category tree) in accordance with the rules.

A rule is created by selecting keywords from a desired article. One rule can take one or more keywords. Rules can be parents to other rules. A parent rule will be more generic rule than its children. Figure 3-3 shows an example of the rules criteria. Rule 0 is the root rule that takes all articles that do not satisfy any of its child rules, which are Rule 1 and Rule 4. Rule 1 has two child rules Rule 2 and Rule 3. Rule 4 has only Rule 5.

Figure 3-4 shows an example of an article retrieved by the MCRDR System. Firstly, Rule 1 and Rule 4 will be applied to the article at the same time. In this case, Rule 4 is not satisfied while Rule 1 is satisfied due to the case that the article has these keywords (Obama and political) and the article initially will be associated with the specified category of Rule 1. The system will then check if there are any child rules to apply to this article. According to Figure 3-3, the system will apply Rule 2 to the article and check for the keyword "Australia". If the article satisfies the rule, the classification category of the Rule will be taken instead of the previous one. At the same time, Rule 3 will

be applied to the article, and in accordance with the example, the article will not satisfy the rule.

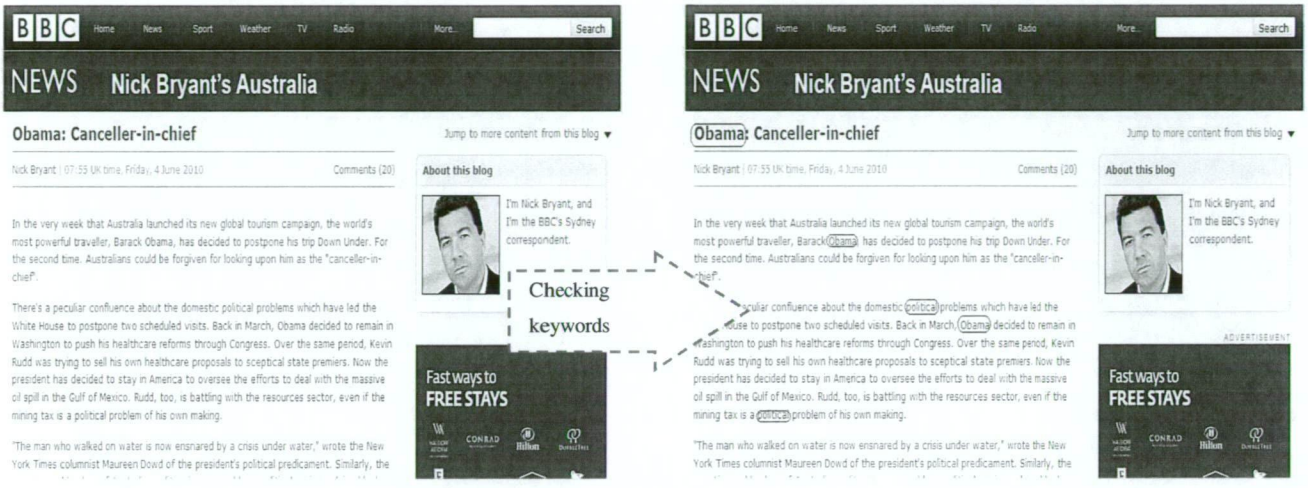


Figure 3-4 the MCRDR system locates keywords (such as Obama and political) that satisfy one rule or more and classifies the article into folders in accordance with what these satisfied rules point to (the article from BBC News)

If neither Rule 2 nor Rule 3 is satisfied, the article will be classified by the MCRDR system in accordance with Rule 1. In real practice, many redundant articles may satisfy the rules and to handle this issue the MCRDR classifier has exceptions to the rule that discard these redundant articles. These exceptions are created when the user selects keywords to create a rule. As soon as the user finishes creating a rule, the MCRDR system will give the retrieved articles that satisfy the rule. The user needs to address what is desired and what is redundant. By removing the redundant articles the system will generate exceptions to the rule. The MCRDR classifier is an incremental system due to the fact that the user can add rules at any time.

3.2.3 Detector Constructor (DC-1)

The detector constructor function in this research checks the classified folders and seeks outliers or abnormal activities to inform the MCRDR classifier about these activities. After that, the MCRDR will alter the scheduling time (visiting time) of the folders that have an outlier to make it sooner than the real scheduled visiting time and obtain more articles.

The Detector Constructor (DC-1) has three processing steps; creating and selecting rules, profiling monitors and weighting to make a decision to inform the MCRDR if an outlier occurs or not. In the first step, DC-1 will gather information from the MCRDR classifier about the classification folders, the installed sites in the system and the delay time used for visiting these installed sites.

After collecting these data, the DC-1 will automatically create all the rules that will be used for the next steps. The rules are used as conditions, which represent the number of retrieved articles between the current and previous visiting time. The structure of these rules are (Time= currentTime, delayTime=delay, Folder=folderId, Site= siteId). The creating step will create all possible rules that the DC-1 might use for a day time. For example, if the visiting delay is two hours, the folder ID is one and site ID is two, the DC-1 will create twelve rules for this site in this folder in accordance with the twenty-four hour for a day divided by the delay time, which is two in this case.

In addition, depending on how many sites and folders that the MCRDR system has, each site will generate twelve rules for each classified folder if the delay time is two hours.

After creating all the possible rules, the system calls the selecting processes to gather the reasonable rules that may be satisfied in the profiling step. The selecting strategy is to obtaining the current time and comparing it with the visiting time of the rules. For example, if the current time is 2:00, the DC-1 will send all rules that have the same visiting time to the profiling monitors. This step will reduce the processing time due the fact that it sends suitable rules instead of sending all the created rules in each checking time, most of which clearly will not be satisfied because of the different time.

The profiling monitors will examine the rules that are collected from the selecting process. As explained above, there are two monitors that are used in the profiling step; the profile and the use monitors. Both of these monitors use two types of evaluation templates; the thresholds and the standard deviation template. The profile monitor obtains the selected rules and applies these rules to the classified articles in the following steps:

1. The monitor will take a rule and retrieve its information (time, delay, folder and site).
2. From the information, the monitor detects the collecting time, which is between time (the current visiting time) and time – delay that the rule will be tested in. In addition, the monitor

recognizes the classified folder which it will look in and the sites from which the articles come.

3. The monitor evaluates how many articles satisfy the rule.
4. The monitor uses the two templates of the evaluation above:
    a. It keeps track of the maximum threshold on normal activity days.
    b. On these normal activity days, it also calculates the mean and standard deviation, and attaches them to the rule.
5. The monitor will redo steps 1-4 until all the rules that have the same classified folder are finished.
6. The classified folder will be associated with the maximum threshold, which is the sum of the rules' thresholds, and mean and standard deviation.
7. The monitor will redo steps 1-6 until all folders are covered.

The classified folders will have maximum threshold, mean and standard deviation for each visiting time. This strategy is used to detect any event when it occurs. Also, many web sites have different main posting times.

Although the use monitor has similar steps to the profile monitor, it calculates the recent visit outputs only and compares these outputs with the profile monitor outputs to generate the use monitor outputs which will be sent to the weighting step. For instance, if the rule (Time= 2:00, delayTime= 2, Folder=1, Site= 2) is profiled with 15 articles as

maximum threshold. If the use monitor detects more than 15 articles, the use monitor output will be 1 and this will be associated with the rule. It associates 0 to the rule if it is less than the profiling monitor threshold. In the standard deviation template, the rule will be associated with (mean, SD) in the profiling monitor. The use monitor will check if the standard deviation is greater than 0, it will compare the current number of articles with the mean of the rule. If the mean is greater than the current number of articles, 0 will be attached to the rule as an output. Otherwise, the output will be (the number of the articles – mean) divided by the standard deviation. However, if the standard deviation is equal to 0, it means that the rule under the profiling monitors and the output will be the number of the articles.

The most important outputs are the use monitor outputs for the classified folders because from these outputs the system generally will make a strong decision whether there is an outlier or not. For each classified folder, the use monitor compares the current threshold with the maximum threshold. The monitor will apply the standard deviation template to each folder as well. The outputs will be similar to the outputs of the rules.

DC-1 joins the outputs from both the threshold and standard deviation templates of the use monitor. Then, the DC-1 system will decide if there an outlier from these outputs or not. When there is an outlier in one of the classified folders, the DC-1 system will notify the MCRDR to set a revisiting time to all the sites that belong to the outlier folder.

From the use monitor the system should know which sites exceed their maximum threshold and by how many standard deviations are they larger than the mean.

After detecting the event, the revisiting time will be set every 10 minutes until the DC-1 detects no article. Then, the next visiting time will be set as the usual visiting time unless another outlier occurs.

3.2.4 Simulating the System

Data Collection

In order to simulate the dynamic scheduling system, a data set of web pages that is collected regularly is needed. This research used a Java program to collect data and this program employs the MCRDR classifier to classify articles into classification folders.

Twenty-six sites were monitored every 30 minutes from 2nd May to 6th June 2010. Unfortunately, there was no real event in this period but the FIFA World Cup 2010 (International Soccer competition) is about to occur. It is expected that the articles will increase significantly when closes to this event.

Allied to this event, most of the sport sites give more attention to the finals of many soccer competitions in May 2010 such as European Champions League, England Soccer League "the Premier League", Spanish Soccer League "La Liga" and the Italian Soccer league "Serie A". The research makes use of these events and creates a folder for

each expected event. The created folders are World Cup, Champion League, La Liga, Premier League and Serie A. these folders will be used to simulate the scheduling system.

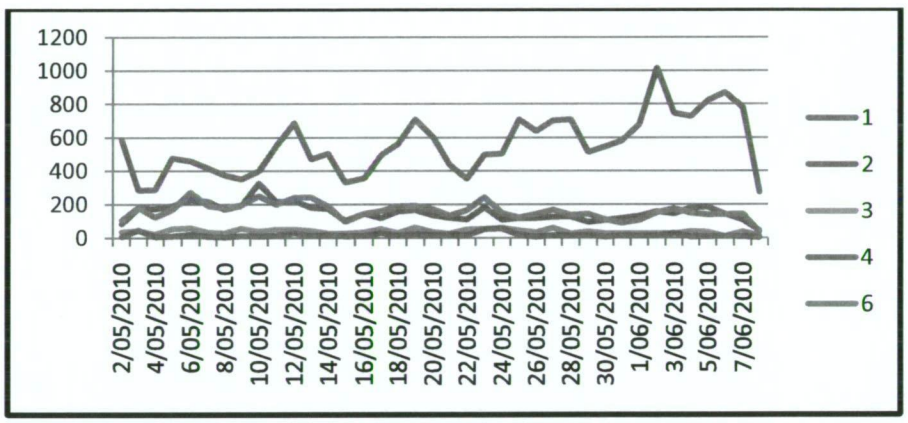


Figure 3-5: number of articles retrieved from 2nd May to 6th June 2010 inside each classified folder. 1: World Cup 2010, 2: Premier League, 3: LA Liga, 4: Serie A and 6: Champion League.

According to Figure 3-5 all classified folders except World Cup 2010 show an ordinary flow of articles with no significant change in the range of daily activity during the collecting period. Therefore, the World Cup will be simulated with the Detector constructor to show how it should work with real data.

Simulating the Scheduling System (DC-1)

As described above, the aim of this dynamic scheduling system is to provide an automatic scheduling strategy for retrieving articles from the web pages which the user is interested in. Also, it detects unusual publishing of articles and sets an earlier retrieving time than that which

is automatically planned as the next retrieving time for these web pages to obtain the least delay between the actual posting time of the articles on the web and when the system collects these articles.

Simulating the classified folders of the system needs to acknowledge the delay time that is used for revisiting web pages to collect articles. For example web sites can be assigned 2, 4, 8, 12 or 24 hours as a regular visiting time. In other words, if the delay time is eight hours the web sites will be visited every eight hours to detect new articles or a change in the web pages.

In this research, the data was collected and classified into folders in advance to test the dynamic scheduling efficiency because merging the two systems into one system will take a long time to implement. The Detector Constructor will be applied to the classified folders and the visiting time of the folders will be rescheduled if there is any outlier in one or more folders.

Figure 3-6 shows how many articles were classified by MCRDR into the World Cup folder during the collection time. To simulate the DC-1 processes, Figure 3-7 shows real data which was gathered on 19th May 2010 for instance. This day was chosen because it shows a significant increase in the activity of the web pages.
With the delay time of two hours the DC-1 visualizes the data as shown in Figure 3-8. The DC-1 collects articles that are posted within the period of time between the previous visit and the current scheduled

visit. Figure 3-9 also simulates the system with four hours as delay time.

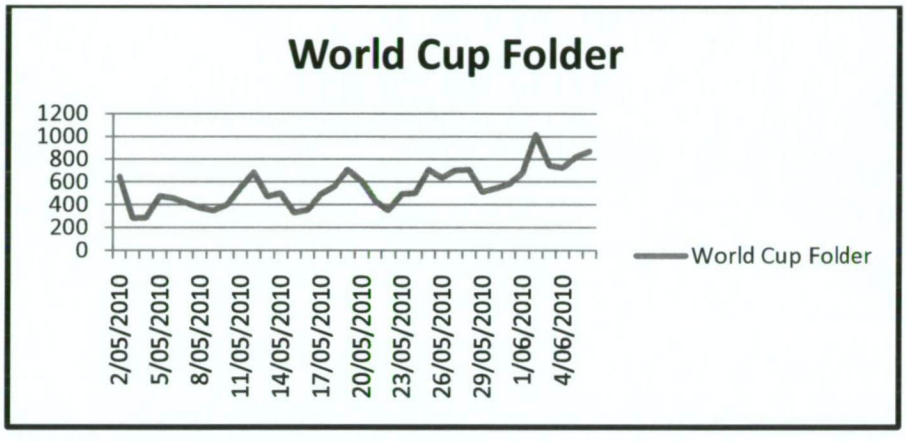


Figure 3-6: Number of articles retrieved from 2nd May to 6th June 2010 in World Cup folder.

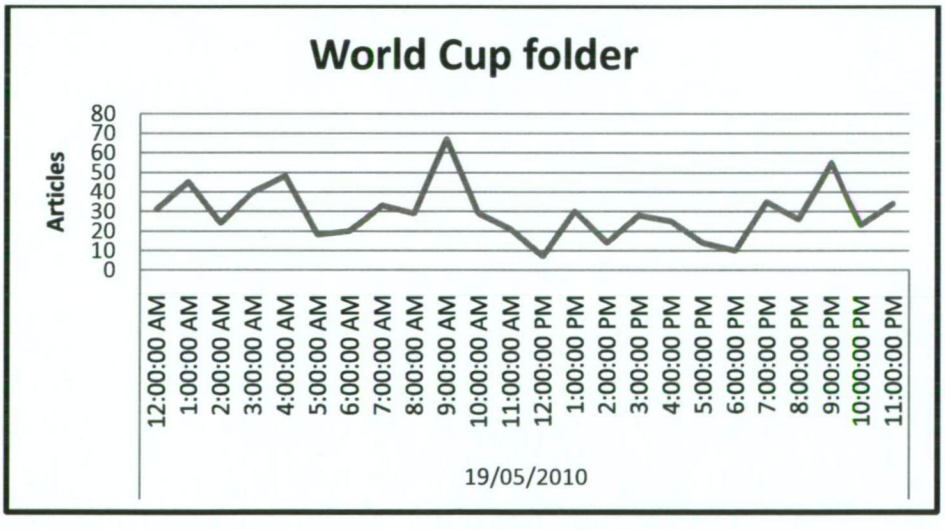


Figure 3-7: Number of articles retrieved at real time on 19th May 2010.

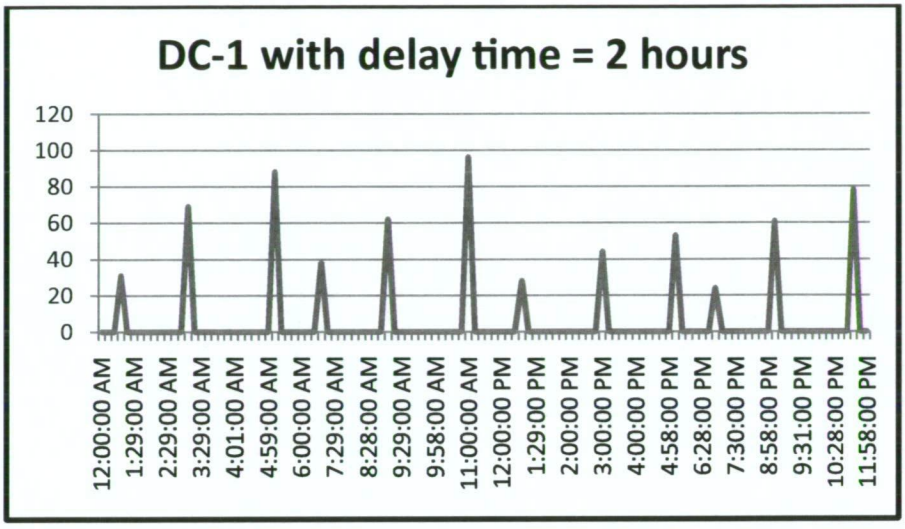


Figure 3-8: Simulation of the Detector Constructor when delay time is equal to two.

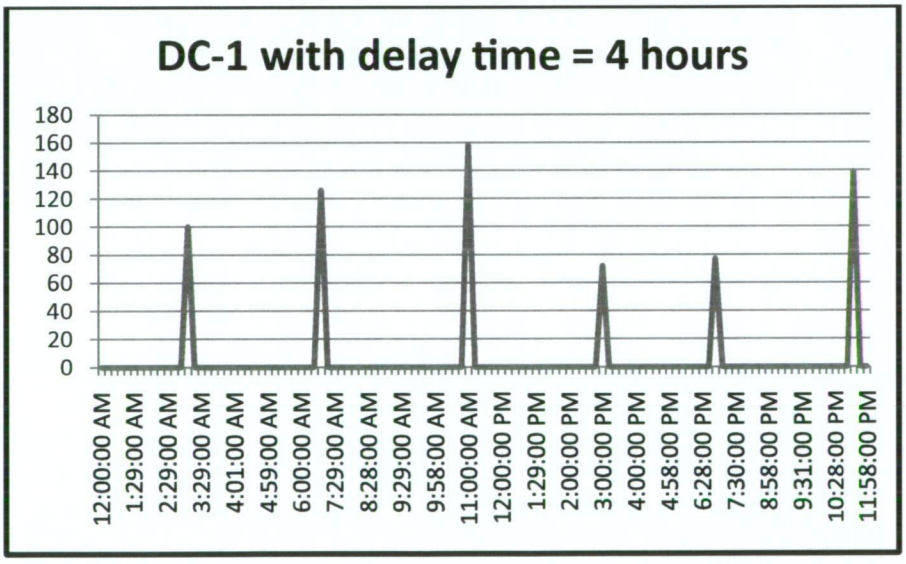


Figure 3-9: Simulation of the Detector Constructor when delay time is equal to four.

According to the Detector constructor system each rule represents a visiting time which has a Mean and Standard Deviation. This is very important because the majority of news web sites and blogs do not have a fixed time for publishing articles. This case can be seen clearly in Figures 3-7, 3-8 and 3-9.

The data indicates that at 11:00 pm (on 19th May 2010) there was unusual posting activity. When the DC-1 system compares the profiled data with the current or use data, the DC-1 will raise an alarm and sets a sooner visiting time to the classified folder as shown in Figure 3-10. The system will continue visiting the folder until it does not detect any article. The next visit will then be postponed to the next scheduled visit, which is a fixed rule.

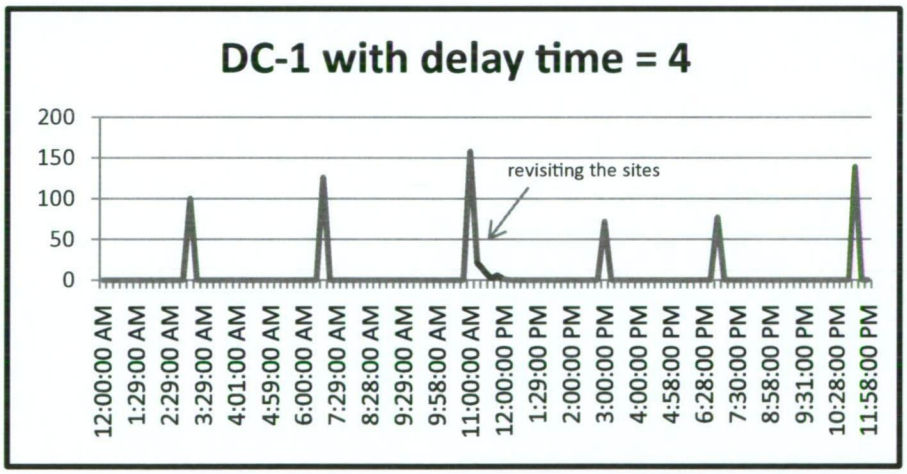


Figure 3-10 : Detecting unusual posting and resetting the visiting schedule of the web sites in this classified folder.

The idea behind the rescheduling is that in many cases, when there is an event, a small range of web sites publish articles faster than others due to many factors such as the location of the event and whether the event is a local or international event.

The DC-1 already has statistical knowledge of the web sites in the classified folders, such as how many articles are posted on each site. The system obtains this knowledge by comparing the rules with the articles in the classified folder to estimate the number of articles satisfying a rule. From these results, the DC-1 system will try to retrieve more articles from the sites that have a low number of articles. Also, it will visit the sites which cause the outlier to see if there more articles to obtain.

4 Findings

On one hand, this research aims to provide a dynamic scheduling system that handles unexpected events. However, nothing unexpected occurred during the data collection time. So, this affected the expected major findings due to the fact that the system did not generate significant results.

On the other hand, Figure 4-1 shows the number of articles in one of the classified folders, World Cup 2010. It can be seen that the number of articles fluctuated. Although Figure 4-1 reveal a range of unusual posting activities which exceed the standard deviation, the topics of these articles should be expected due to the fact that the World Cup

2010 was about to begin, and every national team which participates in the event needs to prepare by having friendly matches.

Also, many articles discuss the history of the teams and some state former soccer players' opinions about the matches. There is no evidence showing that something unexpected occurred such as a famous star being eliminated from his team list due to injury or refusal of participation by any team. Therefore, the system needs to employ an expected threshold that guesses an increase in certain dates. For example, a friendly match between Brazil and Germany should be expected to raise the number of articles up to 70 percent more than the usual posting. If one of these teams wins (5-0) or more, the number of articles may reach 80 percent. The 10 percent difference will be considered as an unexpected event.

When the system deals with these expected posting activities as unexpected events, it stops rescheduling visiting times within a very few visits and a small amount of articles are retrieved. This fact can be seen clearly in Figure 3-10.

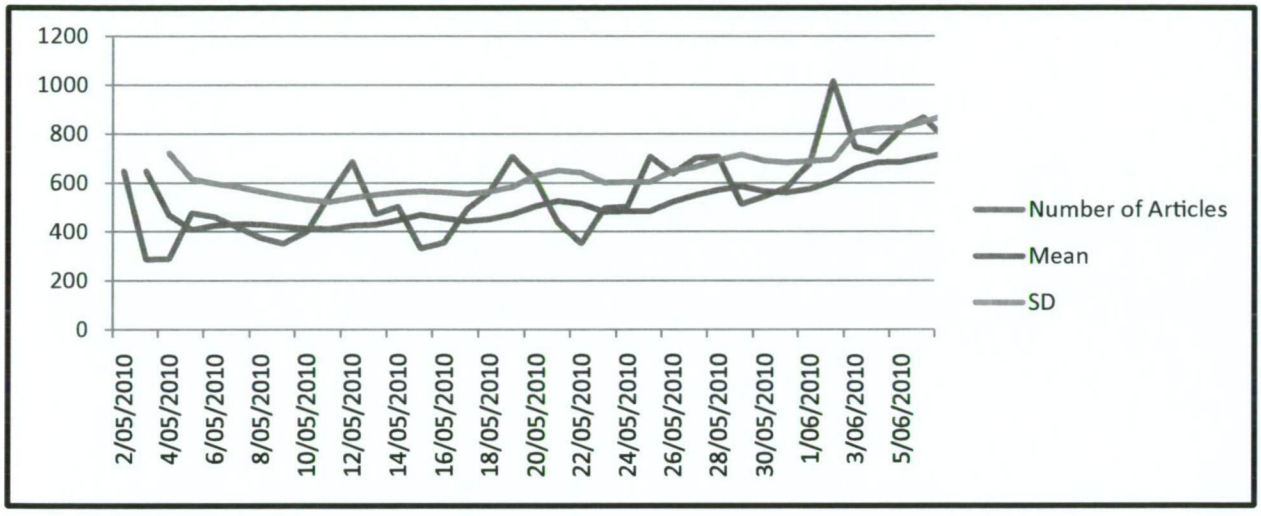


Figure 4-1: World Cup 2010 folder with 24 hours as delay time. Mean and Standard Deviations (SD) are calculated using the previous ten days data.

5 Conclusion and Future Work

In conclusion, this research shows the limitations of using static scheduling approaches which have no mechanism for dealing with unusual or unexpected posting activities (events) and they ignore the fact that most web pages do not have fixed time for publishing. A dynamic scheduling approach is suggested to overcome these limitations. It also provides fewer visiting times to web pages and less delay time between the actual publishing and retrieving time.

To achieve these dynamic scheduling goals, this research combains two rule-based approaches. The first approach is MCRDR, which is used as knowledge based system that retrieves articles and classifies them into classification

folders. Further, it has a mechanism to reject noisy articles where each rule can have exceptions.

The other rule-based approach is DC-1, which is used as an outlier detector to send alarms and make absolute decisions about whether an event occurs or not. It is an adaptive system due to the fact it has a profiling monitors. The dynamic scheduling strategy relies on DC-1 decisions.

Unfortunately, DC-1 was not fully implemented because of many issues. One of these issues is that DC-1 was suggested in the last month of the research time. The first choice was a statistical model of outlier detection models called Grubbs' method. Although it is simple, its adaption to changes in the daily flow of articles is limited.

As a result of time constrains, the DC-1 was simulated with real data in accordance with this short limit of time. It was an unfortunate luck for the research that no unexpected events occurred during the data collection time. However, this unfortunate luck reveled a significant finding that negatively affects the efficiency of the system. The finding is that the system needs to have an algorithm that estimates the "expected threshold" for expected events, such as a major national meeting or matches between skillful and famous teams.

Study into an adequate approach that can estimate the expected threshold can be considered as future work in this area.

6 References:

1. Arning, A Agrawal, R & Raghavan, P 1996,' A Linear Method for Deviation Detection in Large Databases', *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp 164–169.
2. Bishop, C M 1994, 'Novelty detection & Neural Network validation', *Proceedings of the IEE Conference on Vision, Image and Signal Processing*, pp 217–222.
3. Bright, L, Gal, A & Raschid, L 2006, 'Adaptive pull-based policies for wide area data delivery', *ACM Transactions on Database Systems (TODS)*, vol. 31, no. 2, pp. 631 - 671.
4. Davis, R 1986, 'Knowledge-Based Systems', *Science*, vol. 231, no. 4741, pp. 957-963.
5. Everts, TJ, Park, SS & Kang, BH 2006 'Using formal concept analysis with an incremental knowledge acquisition system for web document management', paper presented to ACM International Conference Proceeding Series.
6. Fawcett, T & Provost, F 1997, 'Adaptive Fraud Detection', *Data Min. Knowl. Discov.*, vol. 1, no. 3, pp. 291-316.
7. ---- 1999, 'Activity monitoring: noticing interesting changes in behavior', *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, San Diego, California, United States.
8. Hendriks, PHJ & Vriens, DJ 1999, 'Knowledge-based systems and knowledge management: Friends or foes?' *Information & Management*, vol. 35, no. 2, pp. 113-125.
9. Hodge, V & Austin, J 2004, 'A Survey of Outlier Detection Methodologies', *Artif. Intell. Rev.*, vol. 22, no. 2, pp. 85-126.
10. Kang, BH Compton, P & Preston, P 1995, 'Multiple Classification Ripple Down Rules: Evaluation and Possibilitie', paper presented to Possibilities Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop Banff.

11. Kang, BH, Kim, YS, Kang, SW & Compton, P 2009, 'Using Knowledge Base for Event-Driven Scheduling of Web Monitoring System'.
12. Pandey, S, Dhamdhere, K & Olston, C 2004, 'WIC: a general-purpose algorithm for monitoring web information sources', *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, VLDB Endowment, Toronto, Canada.
13. Pandey, S, Ramamritham, K & Chakrabarti, S 2003, 'Monitoring the dynamic web to respond to continuous queries', paper presented to Proceedings of the 12th international conference on World Wide Web Budapest, Hungary
14. Stolfo, S, Prodromidis, AL, Tselepis, S, Lee, W, Fan, DW & Chan, PK 1997, 'JAM: Java Agents for Meta-Learning over Distributed Databases', paper presented to Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.